Bellingham OS-9 Users Group

Gimix, CoCo, Atari, Mac
6809 - 68K OS-9 Level 1, 2, 3

.ume I No. 4 March 30,1990

OS-9 MEETINS:

Meetings are held at 7:30 p.m., the second Thursday of each month in room 109 at Sehome High School.

BENEFITS TO MEMBERS:

As a participating member of our new Bellingham OS9 Users Group you enjoy many benefits:

1. Newsletter
2. OS9 Bulletins
3. Public Domain Library
4. Technical help
5. Lectures and demonstrations
6. Periodic group purchases
7. Membership List
8. Access to GIMIX Level-III OS9

HELP WANTED!

Our group needs editorial volunteers. If you can contribute with information or helpful experiences of your own, please contact Rodger Alexander. The health of our newsletter depends on contributions made by many members of our group.

N THIS ISSUE:

SUBSCRIPTION INFORMATION:

Newsletters are available free to those in attendance at the monthly meetings. If you would like to receive the newsletter in advance by mail a subscription rate of $3 for 6 monthly issues or $6 for 12 monthly issues is available.

Contact: Rodger Alexander
3404 Illinois Lane
Bellingham, WA 98226
(206) 734-5806
Delphi: "SALZARD

1-MEG of Ram

A little week-early birthday present showed up at my doorthis Saturday (3/3) -- my 1 Meg RAM upgrade kit from CRC/Disto. I've been too busy upgrading some programs to do any more than look it over, but here's what I've seen so far: There are three hardware pieces: the centipede gadget you solder straddling your 6809 CPU, a garden-variety 512K RAM upgrade board (with socketed 120 ns chips crammed into a really compact layout), and an equally tiny all-soldered board of TTY logic that implements the 1 Meg switching. About a dozen chips on that last item, looks very professional. A fourth hardware piece is a big plug-in DC power supply, to take the load off your poor little Coco3 supply. Included is an OS9 disk of IPatch files (and IPatch!) to zap GrfDrv and VDGInt into understanding 1 Meg. Would you believe the GrfDrv patch also includes all of Kevin Darling's speedups. Good, since you must start from "virgin" Tandy modules to ipatch. I might not install the beast until after RainbowFest, but I'm really looking forward to editing and compiling C while Umuse3 is running. Heck, I can even keep a grafix window open all the time...might even dust off MultiVue. (Just to test the upgrade, you understand). When you boot up with the upgrade, you have to execute a little machine-code file "mega" to enable the 1 Meg. This is so you can run certain games (King's Quest, etc) in 512K; these games tweedle the GIME/DAT regs directly (slap! slap the wrist!) and so have to run in "native" mode. There's also a restriction that one video window can't straddle the boundary between hi and lo 512 halves. Dunno whether the patched GrfDrv and VDGInt "know" not to do that, or Tony justassumes that your screens, which OS9 allocates from top of RAM downwards, won't take up over 512K. That's 16 screens of 640x4 or 320x16 colors, so may be safe most of the time.

the F.C.C.

Once again the FCC rears it's head in an attempt to wring more money out of us modem users:

BEWARE!!!

A new regulation that the FCC is quietly working on will directly affect you as the user of a computer and modem. The FCC proposes that users of modems should pay extra charges for use of the public telephone network which carry their data.

In addition, computer network services such as CompuServ, Tymnet, & Telenet would also be charged as much as $6.00 per hour per user for use of the public telephone network. These charges would very likely be passed on to the subscribers.

The money is to be collected and given to the telephone company in an effort to raise funds lost to deregulation. Jim Eason of KGO newstalk radio (San Francisco, Ca) commented on the proposal during his afternoon radio program during which, he said he learned of the new regulation in an article in the New York Times.

Jim took the time to gather the addresses which are given below. What you should do: First, find three or more BBS systems which are not carrying this message and

upload this text. Then, print three copies of the letter which follows (or write your own) and send a signed copy to the three addresses. It is important that you act now. The bureaucrats already have it in there mind that modem users should subsidize the phone company and are now listening to public comment.

Please stand up and make it clear that we will not stand for any government restriction on the free exchange of information.

The three addresses to write to: (a letter to send follows)

Chairman of the FCC
1919 M Street N.W.
Washington, D.C. 20554

Chairman,
Senate Communication Subcommittee
SH-227 Hart Building
Washington, D.C. 20510

Chairman,
HouseTelecommunicationSubcommittee
B-331 Rayburn
Building, Washington, D.C. 20515

Dear Sir,

Please allow me to express my displeasure with the FCC proposal which would authorize a surcharge for the use of modems on the telephone network. This regulation is nothing less than an attempt to restrict the free exchange of information among the growing number of computer users. Calls placed using modems require no special telephone company equipment, and users of modems pay the phone company for use of the network in the form of a monthly bill. In short, a modem call is the same as a voice call and therefore should not be subject to any additional regulation. (or something similar)

OS9 Level-II (upgrade)

by Kevin Darling

Suffice to say that it was an o**icial upgrade, that it took a long time to finish, and then (as everyone here should know quite well by now) Tandy stopped buying CoCo software. That left things in limbo. It means weird deals must be worked out all around. This takes time, patience, and diplomacy. They're convinced that the parties involved will figure out something tho, and that's why the interest must be kept simmering (in case they DO end up needing a letter campaign - heh heh). Those who don't want to hear about it until it's officially out, should just close their eyes <smile>.

What's improved in it? That was the subject of many msgs on the major forums. Too much to cover. Basically: bugs gone, interrupts fixed in software, speed improved, CC3io broken into additional modules JoyDrv, KeyDrv, SndDrv (thus allowing serial mice drivers, new keybd drivers, etc), new calls for sliders and buttons, and some other junk. The list is pretty long. It's very solid.

EDITOR'S NOTE: In the April issue of the Rainbow, Dale Pucket in his "Kissable OS9" article, printed his old familar "FINANCE" program modified to take advantage of the new GFX2 module contained in the not yet released OS9 Level-II upgrade.

It was my fault. He'd been dying to write something about it for a long time; and altho Rainbow had misgivings about printing the article, I told them there was a good chance the upgrade would be available by

the time the magazine came out... or if it wasn't, then people would still love to see and study the code anyway, judging from all the questions on networks. Seemed like it would be a special gift similar to the fast grfdrv patches.

Obviously I was wrong <wry smile>.

It's very clear now that we should never have mentioned the upgrade. Another lesson learned the hard way, eh?

Homework

by Rodger Alexander

QUICK REVIEW: So far we have used several OS9 commands to create a directory (MAKDIR) and then transferred files to our new directory using the COPY utility. Using the BUILD utility we wrote a text file and last month we used the EDIT utility to make modifications to our previous text files. By now you should be getting pretty swift at jumping around between directories.

LESSON #3:
The "Startup" file is a procedure file (also known as a "script file" [also known as a "batch file" to MS-Dos users]). Basically it's a text file containing a list of system or DOS commands that you want the computer to execute. A simple "Startup" file might look like this:

```
display 0c
setime </term
list /dd/sys/motd
```

"Display", "Setime", "List" are three executable system commands that you could just as well have entered from the computer and get the same results, but by putting them in a text file the computer "reads" the instructions and executes them without you having to type in the same instruction everytime you boot up OS9.

OK! Now lets get to work and create our own startup file using the BUILD and EDIT utilities:

1. Enter: BUILD startup
2. At the "?" prompts, enter the following listing:

```
Display 1b 33 01 1b 32 00 1b 34 01 C
List /d0/sys/motd
Setime </term
List /d0/sys/initialize
Load /d0/cmds/utilities
Iniz w7
Shell i=/w7&
```

Press [ENTER] by itself to save your "startup" file and end the BUILD program, returning you to the OS9 shell.

NOTE: To make our "startup" file work we are also going to have to create a "MOTD" (Message Of The Day) in our SYS directory, an "INITIALIZE" (initializing or "loading files" sign) in our SYS directory, and merge some useful command utilities that we would like to have loaded into memory.

3. Enter: CHD SYS;DIR

NOTE: You are changing the default data directory from root directory (/d0) to the SYS (/d0/SYS) directory. The semicolon (;) serves as a deliminator so that you can append a second command to the first. In this case the second command is DIR.

4. Enter: BUILD motd
5. At the "?" prompts, enter the following listing:

Enter: 24 spaces, then 33 "*"(stars/asteriks)

Enter: 19 spaces, then 7 "*", then 29 spaces, then 7 "*"
Enter: 17 spaces, then 7 "*", then 5 spaces, then "WELCOME TO OS9 LEVEL II" then 5 spaces, then 7 "*"
Enter: 17 spaces, then 7 "*", then 5 spaces, then 23 "=", then 5 spaces, then 7 "*"
Enter: 19 spaces, then 7 "*", then 29 spaces, then 7 "*"
Enter: 24 spaces, then 33 "*"
Enter: [ENTER] (Press the ENTER key)

NOTE: The above entry assumes an 80 column screen. If your system comes up on a 40 column screen, replace the first number entry on each line by 1/2 (24/2 = 12, 18/2 = 9, etc.)

6. Enter: BUILD initialize
7. At the "?" prompts, enter the following listing:

Enter: 1 space
Enter: 1 space
Enter: 1 space
Enter: 1 space
Enter: 1 space
Enter: 1 space
Enter: 29 spaces, then "I N I T I A L-I Z I N G" (space between each ltr.)
Enter: 1 space
Enter: 1 space
Enter: 1 space
Enter: [ENTER] (Press Enter key)

NOTE: Replace 29 spaces with only 9 if using 40 column screen.

8. CHD /D0/CMDS
9. Enter: MERGE dir mdir copy del list free echo >utilities

NOTE: When OS9 creates a file, it always assumes that it is a "data file". But our newly created utilities file is an "executable file" located in the CMDS directory, but OS9 still thinks it is a "data file" and will not load or execute the file until we change the file attributes codes. Just to prove my point, try loding our merged utilities file:

Enter: LOAD utilities

OS9 should respond wth a #214 Error (No permission). You cannot load a "data file".

Enter: ATTR utilites

OS9 should respond with "—r-wr". The "R"s tell OS9 that this file may be "read" and the "W" tells OS9 that this file may be written to, but an "E" must exist in this attribute code in order for OS9 to understand that utilities is an executable file. So.....

10. Enter: ATTR utilities e
(OS9 should respond with "—rewr")

I didn't want to get into file attributes, but as you can see, OS9 made me do it!

We now have all that is required by our "startup" file, however, our "startup" file assumes that we are booting up on drive 0. If we were using a hard drive, the startup file would fail to function due to the "/d0" references. We need to change the "/d0" to "/dd" (Default Directory). Remember last month when we used the "C" (Change command) to modify our original HOMEWORK text file? The macro capability of the EDIT command can make these changes more easily than editing each line by hand.

11. Enter: CHD /D0
12. Enter: BUILD modify
13. At the "?" prompts, enter the following listing:

c*
./d0./dd.
q

[ENTER] (Press the ENTER key)

That's our macro file. Almost too simple! In the first line the program will be instructing the EDITor to "C"hange "*" (all) occurances of "/d0" and replace with "/dd".

OK, Lets do it:

14. Enter: EDIT startup <modify
(OS9 should list out our startup file with the new changes.....Well, did it?)

CONCLUSION: Now re-boot the computer, and.....if everything went alright, you should have a fancy looking OS9 Boot.

QUESTION: What does the DISPLAY command in our startup file do?

PC to CoCo Connection

by Craig DuBois

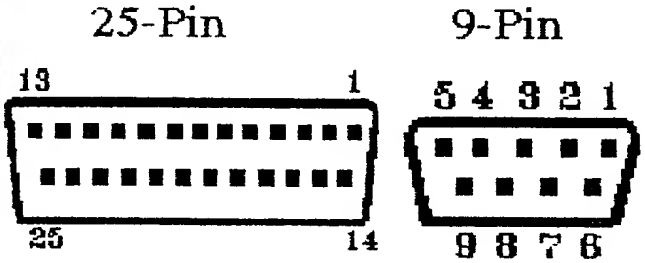
Last week I undertook a new project of debugging a processor controlled system at work. The Audichron IIS takes commands from a telephone office and translates them into pre-programmed custom messages: "The number you have dialed, xxx-xxxx, has been changed to yyy-yyyy", etc. After lengthy phone conversations with the manufacturer and other "experts", I determined a fresh software load was needed.

A Toshiba T1100 laptop PC was sent to me along with several 3 1/2 inch diskettes. With an "expert" on the phone wedged between my shoulder and ear and the PC plugged into the Audichron, I proceeded to load away. After several attempts, the mission was successful.

"But what did I do, and how did it work?" I asked myself. To answer, I took the Toshiba home where I could play in a non-destructive environment. I found the RS232 cable on the Toshiba matched the Deluxe RS-232 Program Pack jack sticking out of slot 1 of my CoCo multi pack. Roger Alexander suggested the null modem which proved to be the finishing touch to the PC-to-CoCo connection. I loaded "DeskMate telecom" in the CoCo and "Cross-talk" in the PC. To my own amazement, the two got along quite nicely, thank you. I even swapped a few data files back and fourth.

So, next time the Audichron decides to get finnicky, it will get a data transfusion via CoCo OS-9. The lone Toshiba can stay in the hands of some "expert" far far away.

RS-232



1 ——CARRIER DETECT—— 8
2 ——RECEIVE DATA—— 3
3 ——SEND DATA—— 2
4 —DATA TERM READY— 20
5 ——GROUND—— 7
6 ——DATA SET READY—— 6
7 ——REQUEST TO SEND—— 4
8 ——CLEAR TO SEND—— 5
9 ——RING INDICATOR—— 22

This is a comparison of the 25 pin serial jack on the CoCo RS-232 program pak and the 9 pin serial jack on the Toshiba T-1100 P.C. With the above pin-out information, a cable can be constructed with a 25-pin plug on one end and a 9-pin on the other end. With such a "converter" cable, the CoCo can be connected to a 9-pin P.C. seral jack.

Windowmaker
Shell+ scriptfile

WINDOWMAKER is a "scriptfile" just like your Startup file for OS9 Level-II with **Shell+**. WINDOWMAKER prompts the user for the Window Number and then prompts for the "type" of window desired. Notice the "VAR"iable and "Prompt" functions. Shell+ is avail-able on DELPHI, COMPUSERV, BASE ACCUMULATOR (455-3410), BBQ-RiBBS (676-5787), and DATA WAREHOUSE (509-325-6787).

```
cls
echo
echo WMAKER
echo window maker with typeset
echo by Ken Heist
merge /h0/sys/stdpats_4 /h0/sys/stdptrs
      /h0/sys/stdfonts >/w
echo
echo
echo TYPES:
echo 1   40x24  text    8c  2Kmem.
echo 2   80x24  text    8c  4Kmem.
echo 5   640x192 graphic  2c  16Kmem.
echo 6   320x192 graphic  4c  16Kmem.
echo 7   640x192 graphic  4c  32Kmem.
echo 8   320x192 graphic  16c  32Kmem.
echo
Prompt Choose Window (1-15):
var.0
Prompt What Type:
var.1
if %1=1
   iniz w%0
   display 1b 20 1 0 0 28 18 0 1 1 >/w%0
   display 1b 31 1 18 1b 32 0 1b 33 1 1b 34 1
       >/w%0
   shell i=/w%0&
   echo  Type1    40x24 Text 8c 2Kmem.
       >/w%0
else
if %1=2
   iniz w%0
   display 1b 20 2 0 0 50 18 0 1 1 >/w%0
   display 1b 31 1 18 1b 32 0 1b 33 1 1b 34 1
       >/w%0
   shell i=/w%0&
   echo  Type2    80x24 Text 8c 4Kmem.
       >/w%0
else
if %1=5
   iniz w%0
   display 1b 20 5 0 0 50 18 0 1 2 >/w%0
   display 1b 3a c8 01 >/w%0
   display 1b 32 0 1b 33 1 1b 34 2 >/w%0
   shell i=/w%0&
   echo  Type5    640x192 Graphic 2c
       16Kmem.   >/w%0
else
if %1=6
   iniz w%0
   display 1b 20 6 0 0 28 18 0 1 2 >/w%0
   display 1b 3a c8 01 >/w%0
   display 1b 32 0 1b 33 1 1b 34 2 >/w%0
   shell i=/w%0&
   echo  Type6    320x192 Graphic
       4c 16Kmem. >/w%0
else
if %1=7
   iniz w%0
   display 1b 20 7 0 0 50 18 0 1 2 >/w%0
   display 1b 3a c8 01 >/w%0
   display 1b 32 0 1b 33 1 1b 34 1 >/w%0
   shell i=/w%0&
   echo  Type7    640x192 Graphic
       4c 32Kmem. >/w%0
else
if %1=8
   iniz w%0
   display 1b 20 8 0 0 28 18 0 1 2 >/w%0
   display 1b 3a c8 01 >/w%0
   display 1b 32 0 1b 33 1 1b 34 2 >/w%0
   shell i=/w%0&
   echo  Type8    320x192 Graphic
       16c 32Kmem. >/w%0
else
endif
clrif
```

RBF & SCF Drivers

by Mike Pleas

OS9 LEVEL-II Basics:

When first starting out in OS9 it's sometimes very difficult to grasp the subtler basics making up this operating system. Unless you've had experience in programming in UNIX or MS-DOS, OS9 can be a very bewildering system to cut your teeth on, as I did. I'm going to attempt to show you the basic inner workings of OS9 and maybe clarify some of the more difficult nuances of our system.

First off, even though the Coco3 has been vastly improved, ie. better graphics and 512k memory; it still uses the 6809 cpu. What this means is that it can only use 64k at a time. The way Tandy got around this was to add a Memory Management Unit (MMU). This piece of hardware works in conjunction with the CPU to pull in and put back from high memory 8k blocks. In other words the active part of memory, the "work space", is constantly pulling in and pushing out information from the relatively inactive parts of memory. Think of it as a subway station where the cars are constantly disembarking and embarking passengers. But this station has limited space so if too many attempt to get off, then an equal or greater number must be transported away.

When you first boot up, OS9 erases all of the Radio Shack Basic rom routines. It then goes through it's first pass in which it sets up spaces, in memory, for the various modules needed to operate. Modules, in a loose sense, can be consideredas rom routines.

In the second pass, OS9 then sets up the parameters necessary to communicate with the various devices. More on that later.

Now we get to a very important aspect in system theory that we tend to ignore. Tables. In order for OS9 to run it must have tables to reference to. MMU, for instance, must be able to know where a particular module is located outside of the work space. Another example is the information the system needs to operate and communicate with the devices. The list of tables that OS9 uses is very long and beyond the scope of this paper.

A brief note to you more advanced programmers. One common misconception is that OS9 always looks up parameters for various routines and devices in the descriptor modules. This is too time consuming and cumbersome. So instead, when OS9 first boots up, Os9p2 passes these parameters to tables in the path descriptor sections within the various device or system managers. For the most part, the discriptor modules are only needed as an occasional reference, or for a warm reset. This is why debugging a descriptor module will show no change after demoding, xmoding, tmoding, etc.

Under no circumstances should any table be altered directly. This can lead to disastrous results. I tried messing with the MMU addresses and succeeded to lock up the computer. I can imagine doing permanent damage to your files if you fool with disk drive tables.

Back to the basics. The modules found in the boot file fall into basic catagories. I've outlined them in block form in the diagram below. We're only concerned with two, RBF and SCF.

RBF (Random Block File)- This includes

all the disk drives. In brief, the way it works is that when you or the system requests to write to or read any part of a disk, RBF looks up in a table (usually on the disk itself) the information needed. The random block part means any part of a disk can be accessed without going through every part before it to get to it.

SCF (Sequential Character File)- This covers just about every other peripheral used- from the screen, to modems, to the speech sound pack. Another term you could use is Asynchronous Serial or Parallel Interface Manager. But we'll stick with SCF.

SCF generally uses a modified RS232 protocal to communicate with it's devices. Thus all information is sent and received in a sequential manner. For example, the screen is sent a series of characters to fill the screen from left to right and from top to bottom and then started over again when a change is made.

That just about wraps up this article. Next month I'll get into some of the commands that you can use to modify some of the modules mentioned above.

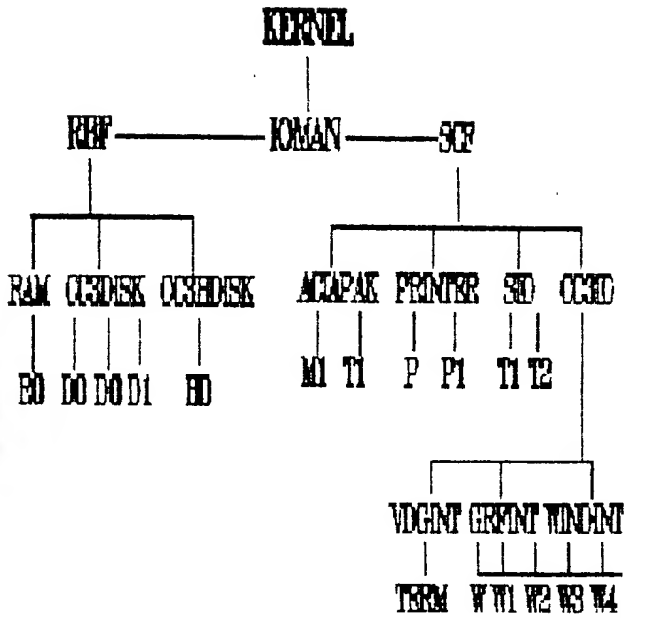
FHL's *Tom Cat*

by Kevin Darling

Don't read this if you don't like to hear about things before they are available <smile>!!! It'll be another month before production begins, I think.

Here's a highly edited version (drastically shortened) of the announcement that FHL just posted to CIS on his version of the KMA [some notes are mine]:

"The TC9 TOMCAT (TM) is a major improvement over the CoCo 3."



* Is over 25% faster! [6309 boosted at times to double speed]
* Uses a PC compatible keyboard.
* Has two 'real' serial ports.
* Supports a serial mouse.
* Has a parallel printer port.
* 512K on board RAM or it can use a CoCo 3 512K memory upgrade.
* Can be upgraded to 1 megabyte with the Disto 1 Meg upgrade [plug in].
* Has 8 bit D to A and A to D. [sound/ joystick].
* Supports an internal speaker.
* Has the standard CoCo bus so that CoCo cartridges can be used.
* Board can be powered by any PC power supply.
* Will work with most, if not all OS9 software.
* Will have RSDOS compatibility thru 3rd party vendors.
* Is K-Bus compatible [bus not required for TC9 operation]"

FHL goes on to envision the possibility of several TC9's in a K-bus along with 68000-68030 cpus running alongside, with either perhaps using the other cpus for I/O processing in some cases.